



PATENT ABSTRACTS OF JAPAN

(11) Publication number: **07152640 A**(43) Date of publication of application: **16 . 06 . 95**

(51) Int. Cl.

G06F 12/06
G06F 15/163(21) Application number: **05296300**(22) Date of filing: **26 . 11 . 93**(71) Applicant: **HITACHI LTD**(72) Inventor:
YAMAUCHI MASAHIKO
YOSHIZAWA SATOSHI
MURAYAMA HIDEKI
HAYASHI TAKEHISA
KITO AKIRA(54) **DECENTRALIZED COMMON MEMORY SYSTEM**

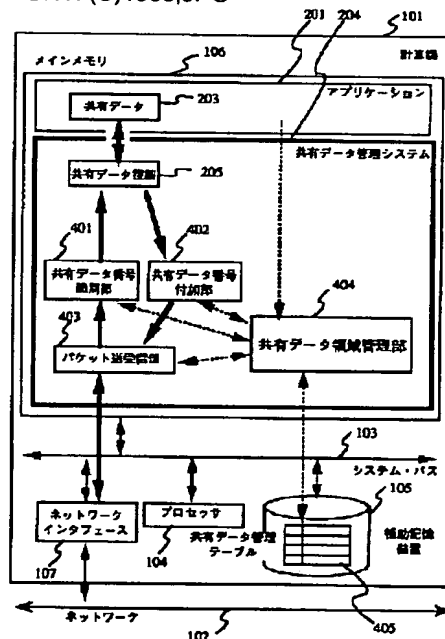
elements.

(57) Abstract:

COPYRIGHT: (C)1995,JPO

PURPOSE: To decrease the amount of transferred data and the frequency of transfer and to effectively utilize network resources by obtaining information regarding common data such as the timing, etc., of the guarantee for the consistency of common data areas and common data as to the common data that application software uses as if one data were shared for the application software.

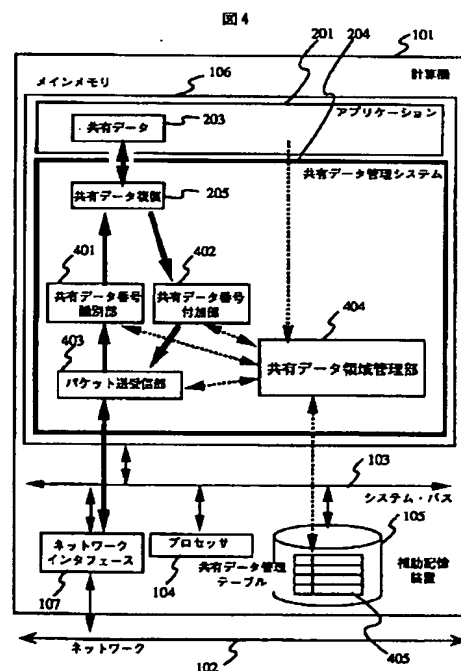
CONSTITUTION: A common data management system 204 consists of a common data number discrimination part 401, a common data number addition part 402, a packet transmission and reception part 403, and a common data area management part 404, and hold respective constituent elements of a common data management table 405. When the application software 201 declares data (common unit) to be shared and mapping to a memory to the common data area management part 404, the correspondence between the common data and memory is stored in the common data management table 405. A computer which indicates common data modification and a computer which receives the common data modification guarantee the consistency of the common data by using the respective constituent



(11)特許出願公開番号

(43)公開日 平成7年(1995)6月16日

3 2 0 M



【特許請求の範囲】

【請求項1】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ方式において、計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共有単位で指定し、指定された共有単位にて共有データに対応付けて管理する共有データ管理手段を上記複数の計算機の少なくともいずれか一つに設けたことを特徴とする分散共有メモリ方式。

【請求項2】請求項1に記載の分散共有メモリ方式において、上記共有データの複製を格納する手段を設け、計算機のアプリケーションは、上記複製に対してRead/Writeを行うことを特徴とする分散共有メモリ方式。

【請求項3】請求項1または2に記載の分散共有メモリ方式において、各計算機間で共有する領域間の一貫性保証を、任意のアドレス及びサイズよりなる共有単位で指定することを特徴とする分散共有メモリ方式。

【請求項4】請求項1または2に記載の分散共有メモリ方式において、各計算機間で共有する領域の任意の部位を一次元配列への写像として指定し、その一次元配列を共有単位として指定することを特徴とする分散共有メモリ方式。

【請求項5】請求項4に記載の分散共有メモリ方式において、上記一次元配列を格納する領域を、共有単位が指定された際に、メモリ上で確保することを特徴とする分散共有メモリ方式。

【請求項6】請求項3に記載の分散共有メモリ方式において、各計算機間で共有する複数の共有単位について、共有単位間での階層関係を定義し、上記一貫性保証を指定した共有単位に加え、その下位階層として定義された共有単位についても一貫性保証を一括して指定することを特徴とする分散共有メモリ方式。

【請求項7】請求項6に記載の分散共有メモリ方式において、上記一貫性保証を行なう際に、指定された共有単位の下位階層として定義された共有単位から、任意の下位階層共有単位を選択して、一貫性保証を一括して行なうことを特徴とする分散共有メモリ方式。

【請求項8】請求項3に記載の分散共有メモリ方式において、各計算機間でデータを交換する手段として、上記一貫性保証した共有単位のデータ交換手段と共に、他のデータ交換手段を備え、交換するデータのメモリ上での配置に応じてデータ交換手段を選択することを特徴とする分散共有メモリ方式。

【請求項9】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ方式において、計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共

有単位で指定し、指定された共有単位にて共有データに対応付けて管理するための共有データ管理テーブルと、上記共有データ管理テーブルを管理する共有データ領域管理手段と、

他の計算機とのバケットによる上記共有データを送受信するバケット送受信手段と、

上記バケット送受信手段を介して送信するバケットを生成するために、共有データを格納するメモリ領域を調べ、そのメモリ領域に対して共有データ番号を付与する共有データ番号付与手段と、

10 上記バケット送受信手段を介して受信したバケット中の共有データ番号を識別し、共有データ番号に対応するメモリ領域を上記共有データ管理テーブルを用いて調べ、対応するメモリ領域に共有データを書き込む共有データ番号識別手段と、

計算機のアプリケーションからRead/Writeを行うために共有データの複製を格納する手段とを、上記複数の計算機のうち少なくともいずれか一つに設けたことを特徴とする分散共有メモリ方式。

20 【請求項10】請求項9に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、共有データ番号を格納する共有データ番号エン트리と、共有データを管理している計算機かどうかを示す管理ノードフラグエン트리と、共有データの開始アドレスを示す共有データアドレスエン트리と、共有データの複製の開始アドレスを示す共有データ複製アドレスエン트리と、共有データの大きさを示す共有データサイズエン트리と、共有データを保持している計算機を示す計算機リストエントリを有することを特徴とする分散共有メモリ方式。

30 【請求項11】請求項10に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、アプリケーションが使用する共有データが連続領域か否かを示す分割フラグエン트리と、幾つの分割されたデータから共有データが構成されているかを示す共有データ要素数エン트리と、分割されたデータがどのくらいの間隔で並んでいるかを示す共有データオフセットエントリを有することを特徴とする分散共有メモリ方式。

【請求項12】請求項11に記載の分散共有メモリ方式において、上記共有データ管理テーブルは、共有データが構造体である場合に、構造体の定義を格納する構造体定義エン트리と、構造体のデータ長を格納する構造体サイズエン트리と、複数の共有データに階層関係があることを示す階層要素フラグエン트리と、階層要素の共有データ番号を格納する階層要素共有データ番号エントリを有することを特徴とする分散共有メモリ方式。

【請求項13】複数の計算機がネットワークで接続され、各計算機のメモリ上に分散したデータを各計算機にて共有する分散共有メモリ処理方法において、計算機のアプリケーションから、各計算機のメモリ上に分散したデータを任意のアドレス及びサイズよりなる共

有単位で指定し、
 指定された共有単位にて共有データを対応付けて管理する管理計算機を決定して、上記管理計算機に対して共有データの使用を要求し、
 上記管理計算機では、指定された共有データが使用されているか否かを確認し、
 使用されていない場合には、指定された共有データに関する情報を上記管理計算機に新規に登録し、
 使用されている場合には、新規に共有データの使用を指定した計算機に関する情報を上記管理計算機に登録し、
 その情報を他の計算機に通知し、
 共有データの使用を指定した計算機では、各計算機のメモリ上に分散した共有データを獲得して、共有データの複製を作成する、
 ことを特徴とする分散共有メモリ処理方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、ネットワークで接続された複数の計算機システムで構成されるネットワーク計算機システムに於いて、物理的に分散配置されたメモリ上のデータを、別計算機上で実行中の複数アプリケーション間で共有するための分散共有メモリ技術に係わり、特にネットワーク資源を有効利用し、その結果、分散共有メモリへのアクセス性能を向上させる分散共有メモリ方式に関する。

【0002】

【従来の技術】計算機に於いて、計算機内部でアプリケーションを実行する仮想的な実体をプロセスと呼ぶ。プロセスを複数生成し、単一CPUの演算資源を一定時間毎に切り換えて各プロセスに割り当てることで、CPU台数よりも多いプロセスを並行に動作させることができる。これを並行処理と呼ぶ。一方、複数のCPUを使用して、複数のプロセスを同時に動作させる処理を並列処理と呼ぶ。

【0003】従来、単一CPU構成の計算機に於いて並行処理を行なう時には、プロセス間でのデータ交換の方法として共有メモリを使用することができた。これには例えば、「UNIXシステムコール・プログラミング、Marc J. Rochkind 著、福崎俊博訳、アスキー出版、1987」の299-315頁に示されたシステムVの共有メモリがある。

【0004】並行処理では物理的にプロセスが同時に実行されないために、複数プロセスによって構成されるアプリケーションの実行性能の向上は望めなかった。そこで、ネットワーク計算機システムでは、複数の計算機をネットワークで接続し、複数のCPUを同時に利用してアプリケーションを並列処理することによって、アプリケーション実行性能の向上を図る。

【0005】ネットワーク計算機システムに於ける共有メモリでは、メモリが物理的に分散しているので、各計

算機のメモリ上に格納されたデータ間の一貫性を保証し、あたかも一つのデータを共有しているかの様に、アプリケーションに見せる必要がある。この技術を分散共有メモリと呼ぶ。

【0006】仮想記憶機構をページング機構で実現する計算機では、そのページング機構を拡張して分散共有メモリを実現することができる。即ち、通常ではメモリアクセスの際、アクセスしようとしたデータがメモリ上に存在しないとページフォールトを発生し、補助記憶装置からデータが存在するページをメモリ上にロードする。分散共有メモリでは、ページフォールト発生の際、補助記憶装置からページを取得する代わりに、他計算機からネットワークを介して自分の計算機のメモリ上にロードすることで実現される。

【0007】分散共有メモリの従来技術の例としては、以下のIvyシステム(米・エール大学)、及びDASHシステム(米・スタンフォード大学)を挙げることができる。

【0008】Ivyシステムでは、各ページに所有者(オナ)が存在し、所有者である計算機が、そのページの複製を保持する計算機の名称を管理する。また所有者のみがそのページへ書き込みを行なうことを許されている。ある計算機で読み出しフォールトが発生した時には、所有者からページの複製が送られる。また、書き込みフォールトが発生した時には、書き込みフォールトが発生した計算機が新しい所有者となり、他の計算機上に存在するページの複製は無効化される。

【0009】DASHシステムでは、16バイトのデータを単位として、Ivyシステムと同様な制御を行なっている。

【0010】これらの従来技術の詳細については、「分散オペレーティングシステム UNIXの次にくるもの、前川守・所真理雄・清水謙多郎編、共立出版、1991」の124-141頁に示されている。

【0011】

【発明が解決しようとする課題】分散共有メモリではメモリ内容の一貫性を保つためにネットワークで接続された計算機間でメモリ内容の交換が行なわれる。この時、従来の分散共有メモリ管理方式で課題となるのはデータ転送サイズとデータ転送タイミングである。

【0012】Ivyシステムについては以下の様な課題がある。第1にデータ転送サイズに関して、メモリ内容交換のために計算機間を流れるデータ転送サイズがページであるため、ページサイズ(例えば、4キロバイト)と比較して小さいデータ(例えば、8バイト)を計算機間で共有している場合、計算機間を流れるデータ転送量に無駄が発生する。このため、実質的なネットワーク資源の利用率が下がるという課題がある。

【0013】また、計算機Aと計算機Bの各計算機でアクセス頻度が高い2つの異なるデータが同一ページ上に

10

20

30

40

50

存在する場合に、計算機Aと計算機Bが交互に同じページに対するメモリ書き込みを行なうと、ページが計算機Aと計算機Bの間を移動する。このため、各計算機がメモリアクセスを行なう度に、ネットワークを介した計算機間でのページ転送が発生し、メモリアクセスの性能が下がるという課題がある。

【0014】第2にデータ転送のタイミングに関して、データ転送はメモリへの読み出しや書き込み時に自計算機内に共有データの存在するページがない時にページフォルトを発生して行なっていた。

【0015】そのため計算機Aと計算機Bが同じ共有データに連続して書き込みを行なうと、最悪の場合、各計算機がメモリ書き込みを行なう度にページが計算機間を移動する。ネットワークを介して計算機間の転送を行なうコストは、自分の計算機内のメモリアクセス速度と比較して大きいと、共有データへのメモリアクセスの性能が下がるという課題がある。

【0016】一方DASHシステムは、Ivyシステムの第1の課題の解決を狙ったシステムである。即ち、ページ単位でメモリ内容を共有しているために発生する無駄なデータ転送を無くすために共有の単位を小さくした。その結果、無駄なデータ転送はなくなったが、メモリの広範囲に対してアクセスを行なうとアクセスフォルトが頻繁に発生することとなった。アクセスフォルトが頻繁に発生すると、ネットワークを介して行なう計算機間でのデータ転送時間が顕著になる。DASHシステムでは、これに対して小さいデータの転送時間を短くすることを目的として、特殊化した専用ネットワークを提案している。しかし、この専用ネットワークは汎用ネットワーク・アーキテクチャとは異なるためコストが高いという問題がある。

【0017】更にIvyシステムの第2の課題である、複数計算機が同じ共有データに対して連続してアクセスした場合の性能低下については、DASHシステムに於いても発生し解決されていない。

【0018】上記課題に起因して、従来の分散共有メモリ方式を、典型的な並列処理手法である領域分割型並列処理に適用した場合、ネットワークを介した通信が頻繁に発生し、並列化による処理の高速化率で良好な値を得ることができない。更にポインタ変数を要素とするデータ構造体を複数計算機間で共有する場合等、共有の対象となるデータがメモリ上で散在して格納されることがあり、このような場合にもネットワークを介した通信が頻繁に発生する。

【0019】本発明の目的は、上記課題を解決し、ネットワーク資源の有効活用、及びアクセス性能の高い分散共有メモリ方式を提供することにある。

【0020】

【課題を解決するための手段】 先ず、第1の課題を解決するために、アプリケーションで共有するデータとメモ

リへの写像を設定可能とし、設定した共有単位を記憶しておくための共有データ管理テーブルを有し、その共有データ管理テーブルを管理するための共有データ管理手段を設ける。

【0021】具体的には、アプリケーションが2次元配列の様なデータを2分割し、分割した境界部分のデータを共有する場合には、アプリケーションからその境界部分のデータを共有単位として設定する。共有単位は、メモリ連続領域、メモリ不連続領域（データサイズ一定、データの間隔一定）、またはそれらの組合わせによって指定する。

【0022】次に第2の課題を解決するために、複数の計算機上の共有データの一貫性を保証するタイミングをアプリケーションが仮想的な共有空間からの（1）最新データの獲得指示、共有空間への（2）最新データの反映指示という2つの情報に基づいて決定し、一貫性を保証する単位として設定された共有単位を使用する。

【0023】具体的には、全ての計算機に共有データを持たせ、共有データを変更する計算機に於ける（2）最新データの反映指示のタイミングで全ての共有データの更新を行なう方法、または一つの計算機に共有データを持たせて、共有データを獲得しようとした計算機に於ける（1）最新データ獲得指示のタイミングで共有データを保持している計算機から共有データをネットワークを介して獲得する方法、更に、共有データの信頼性を高めるために、これら2つの方法を組み合わせて一つ以上の計算機で共有データを保持し、危険を分散してもよい。

【0024】

【作用】アプリケーション実行時に、予め共有単位を設定することによって無駄なデータ転送を防ぎ、ネットワーク資源を有効に活用することができる。また、物理的に分散しているデータの一貫性をアプリケーションが必要な時のみ保証することによって、ネットワークを介した計算機間の転送を最小現に抑えてネットワーク資源を有効活用できる。その結果、共有データへのアクセス性能を向上できる。

【0025】

【実施例】図1は、本発明に於けるネットワーク計算機システム100の一実施例を示すブロック図である。

【0026】本図に於いて、ネットワーク計算機システム100は、複数の計算機101をネットワーク102で接続して構成される。計算機101に於いて、103は計算機を構成する各ブロック間のデータ伝送路であるシステム・バス、104はプログラムを実行するプロセッサ、105はプログラムやデータを格納する磁気ディスクなどの補助記憶装置、106はプログラム実行中に該プログラムやデータを格納するためのメインメモリ、107はネットワーク102との入出力を制御するためのネットワーク・インタフェースである。本発明による共有データ管理方式は、例えば計算機101（1）上の

アプリケーションと計算機101(2)上のアプリケーションが並列実行する時に、データを共有する際に適用される。

【0027】図2は、領域分割型並列処理に対して、本発明による分散共有メモリを適用する場合の一実施例を示す図である。

【0028】ここで領域分割型並列処理とは、図2

(a)に示す様なデータ配列を、図2(c)、(d)に示す様に分割して、それぞれを別々の計算機上で並列に処理する形式の並列処理のことである。流体解析問題や構造解析問題等、多くの科学技術計算には、領域分割型並列処理を適用可能である。

【0029】領域分割型並列処理の一例として、二次元熱拡散解析問題の一解法を図2を用いて説明する。本解析手法では、解析対象となる領域を図2(a)に示す様な $n \times n$ 二次元メッシュ状に区切り、配列データ202としてメインメモリ106上に格納して、プロセッサ104が演算処理を行なう。この際、配列データ202の各要素には、各要素の温度値を格納する。演算処理は、図2(a)の二次元配列の全要素に対して、各要素の近傍要素を用いた計算を行ない、それを各要素の一定時間後の値として設定する処理を、予め定められた終了条件が満たされる迄、繰り返し行なう形態をとる。ここで各要素の近傍要素を用いた計算とは、例えば図2に示した様に、配列要素(m, m)の新たな値を計算する際に、その4近傍点、即ち(m, m-1)、(m+1, m)、(m, m+1)、(m-1, m)と、(m, m)自体の値を用いて計算することである。本計算の一例は、配列要素(m, m)の値に対して、各近傍点の値と(m, m)との値の差異を先ず求め、それに熱伝導率に該当する係数を乗じた上で、(m, m)の値に加算するものである。

【0030】図2(c)、(d)に、本解析手法を二台の計算機上で並列に処理する場合の領域分割を示す。上記の様に本解析では各配列要素の演算のために近傍点のみを用いるため、 $x=0$ から $x=(n/2)-2$ 迄の列202(1)、及び $x=(n/2)+1$ から $x=(n-1)$ 迄の列202(2)に関しては、各領域の演算を担当する計算機のみが読み書きできれば良く、各計算機のローカルメモリ106上に格納されていれば良い。それに対して、 $x=(n/2)-1$ から $x=(n/2)$ までの列203(1)及び(2)は、両計算機から読み書きできる必要があり、共有メモリ領域内に格納される必要がある。

【0031】尚、上記の様な解析問題における典型的な問題規模、即ち図2(a)の配列の要素数は、数百行×数百列から数千行×数千列程度のオーダーである。

【0032】以下に説明する様に、本発明の分散共有メモリ方式では、計算機間でのデータ交換を、上記の様な領域分割型並列処理に対しても効率良く行なえる方式を

提供する。

【0033】図3は、同一のアプリケーションを実行する複数のプロセス間に於いてデータを共有する際の共有データ管理方式の一実施例を示すブロック図である。

【0034】本図に於ては、計算機101(1)上のアプリケーション201(1)と、計算機101(2)上のアプリケーション201(2)とがデータを共有している。ネットワーク計算機システム100を構成する各計算機101は物理的に共有したメモリを持っていないために、物理的に分散した共有データ203(1)と203(2)との一貫性をハードウェア的、ソフトウェア的、またはそれらの組み合わせによる手段で保つ必要がある。この一貫性制御によって、計算機101(1)上のアプリケーション201(1)と、計算機101(2)上のアプリケーション201(2)とが、あたかも一つのデータを共有している様に見えることができる。

【0035】本実施例では、共有データの一貫性を保つ仮想的な空間を各計算機上に存在する共有データ管理システム204によって提供する。アプリケーション201が共有データに対してRead/Writeを行う時には、共有データ管理システム204によって一貫性が保証された仮想的な空間との間で行なう。本実施例では具体的には、共有データ複製領域205に格納されたデータを読み書きして行なう。

【0036】本実施例に於ては、全ての計算機101の共有データ管理システム204には、アプリケーション201で使用する共有データ203の複製205が存在する。この複製205は、アプリケーションが共有データを使用することを宣言した時に生成される。これはアプリケーション201から共有データ203を獲得する要求がきた時の処理遅延を最小限にするためである。共有データ管理システム204では、物理的に分散している共有データの複製205に対して一貫性を保証する。即ち、共有データ管理システムは、ある計算機で共有データの複製を変更した場合には、共有データの複製205を保持する全ての計算機101に対して変更を伝達する。

【0037】また、共有データ管理システム204の必要とするメモリ量を節約するために、全ての計算機に共有データの複製を持たずに、アプリケーション201が最新の共有データ反映の指示を行った時に、その指示を行った計算機上に複製を作成する方式をとっても良い。同時に、共有データの複製205を保持する計算機の識別子を他の計算機に通知する。共有データの複製は、ネットワーク計算機システム内の一台、または一台以上の計算機が保持していても良い。

【0038】図4は、本発明による共有データ管理方式の一実施例を示すブロック図である。

【0039】共有データ管理システム204は、共有デ

ータ番号識別部401、共有データ番号付加部402、
 パケット送受信部403、共有データ領域管理部404
 から構成される。また、補助記憶装置105には、共有
 データ管理テーブル405を持つ。共有データ管理テー
 ブル405は、予めメインメモリ106に展開しておく
 ことによって、その参照、更新処理を高速化すること
 ができる。共有データ管理テーブル405の詳細につい
 ては後述する。

【0040】共有データ番号識別部401は、パケット
 送受信部403から送られてくるパケットの共有データ
 番号を識別する。共有データ番号に対応するメモリ領域
 を共有データ領域管理部404を利用して調べ、対応す
 るメモリ領域に対してデータを書き込む処理を行なう。

【0041】共有データ番号付加部402は、共有デー
 タ領域管理部404から共有データ番号とそれに対応す
 るメモリ領域を受け取り、メモリ領域を読み込んで他の
 計算機に共有データの内容を伝達するパケットの生成を
 行なう。

【0042】パケット送受信部403は、ネットワーク
 ・インタフェース107から送られてくるパケットのコ
 マンドを解析し、そのパケットを処理するブロックに振
 り分ける。また、共有データ番号付加部402で作成し
 たパケットを他の計算機に伝達するためにネットワーク
 ・インタフェース107を起動する処理を行なう。

【0043】共有データ領域管理部404は、共有デー
 タ管理テーブル405の管理と共有データ管理システム
 204のブロックを制御する処理を行なう。また、アプ
 リケーション201の共有データに対する処理依頼を受
 け取り、共有データ管理テーブル405に新しいエント
 リを追加したり、削除を行なう。更に、共有データの
 内容を他の計算機に伝達する様に共有データ番号付加部4
 02やパケット送受信部405を制御する。逆に共有デー
 タの内容を他の計算機から受け取る様に共有データ
 番号識別部401やパケット送受信部403を制御す
 る。

【0044】図中401から404に於いて行なわれる
 各処理は、ソフトウェア的、ハードウェア的、またはそ
 の組み合わせによって行なわれるものである。ソフトウ
 ェア的に行なわれる部分の処理に関しては、以下の様に
 実行される。即ち、各ソフトウェア処理の内容を記述し
 たプログラムは補助記憶装置105に格納されており、
 システム・バス103に接続されたメインメモリ106
 に展開した上で、プロセッサ104によって実行され
 る。

【0045】図5は領域分割型並列処理を行う際に、配
 列データを分割した時の共有データのメモリ配置を示す
 一実施例である。

【0046】本図に於ては簡単化のために、6行×6列
 の配列を列方向に2分割して、2台の計算機を利用して
 並列処理することを例に採る。実際にはより要素数の多

い配列を、行方向、及び列方向に分割して、より多数の
 計算機に割り当てて並列処理を行なうことが想定される
 が、本図を用いて説明する内容を直接適用することが可
 能である。

【0047】配列の各要素は8バイト長の倍精度実数と
 する。分割した境界部分の1列を共有データ番号「0」
 を持つ共有データ501、共有データ番号「1」を持つ
 共有データ502として宣言する。511は、6行×3
 列の部分配列のメインメモリ上の配置を示している。5
 11が示す様に共有データ501はメインメモリ上では
 不連続領域として配置されている。

【0048】一方、本発明に於て共有データを宣言した
 時に作成される共有データ複製のメインメモリ上の配置
 の一実施例は、512に示す様に連続領域になってい
 る。511に於ける共有データの配置と512に於ける
 共有データ複製との配置の対応を共有データ管理テー
 ブル405に於いて管理している。領域511と512に
 格納されたデータの対応付けは、共有データ番号をキー
 として行っている。この対応はアプリケーションが51
 1に於ける共有データを使用すると宣言した時に共有デ
 ータ管理テーブル405内に自動的に作成する。これに
 よって、不連続な共有データを一つの識別子（共有デー
 タ番号）で指定でき、かつ不連続なデータを一つにまと
 めることによって無駄なメモリ領域を必要としない。

【0049】図6は、本発明に於ける共有データ管理テ
 ーブル405の一実施例を示し、特に領域分割型並列処
 理に適したテーブル構造を示す図である。

【0050】共有データ管理テーブル405は、共有デー
 タ番号エントリ601、共有データを管理している計
 算機かどうかを示す管理ノードフラグ・エントリ60
 2、共有データの開始アドレスを示す共有データアドレ
 ス・エントリ603、共有データの複製の開始アドレス
 を示す共有データ複製アドレス・エントリ604、共有
 データの大きさを示す共有データサイズ・エントリ60
 5、共有データを保持している計算機リスト・エントリ
 609からなる。

【0051】また、アプリケーションが使用する共有デー
 タが連続領域か否かを示す分割フラグ・エントリ60
 6、幾つの分割されたデータから共有データが構成され
 ているかを示す共有データ要素数エントリ607、分割
 されたデータがどのくらいの間隔で並んでいるかを示す
 共有データオフセット・エントリ608を保持する。こ
 れによって、アプリケーションは連続したメモリ領域の
 共有データだけでなく、一定サイズのデータが一定間隔
 で並んだメモリ領域を共有データとして宣言することが
 可能である。

【0052】更に計算機リスト・エントリ609によっ
 て、共有データ番号で示される共有データを保持する全
 ての計算機を特定することができる。共有データ変更を
 他の計算機に通知するために、計算機リスト・エントリ

609を利用してよい。利用しない場合には、ネットワークに接続されている全ての計算機に通信を行なうブロードキャスト機能を使えばよい。但し、ここで要求されるブロードキャスト機能は、バケット消失が発生しない、各計算機で受信するバケットの順番が変わらないという信頼性を持つ必要がある。

【0053】共有データの複製を全ての計算機が保持しない場合には、共有データの複製を保持している計算機の識別子を格納する項目を必要とする。

【0054】図7は、複数計算期間で、ポインタ変数を含むデータ構造体を共有する際の共有データのメモリ配置を示す一実施例である。

【0055】図7(a)にC言語でのデータ構造体の宣言の一例を示す。本例に於ては三番目の要素cが正数型のポインタ変数となっている。以下、本構造体を共有する場合を例に、本発明による一実施例を示す。

【0056】図7(a)に示したデータ構造体を共有する場合、メモリ上では図7(b)の511に示す様に配置される。本配置に於て、要素cの領域701に格納されるのはメモリ上のアドレス値であり、これによりポインタを介してデータ領域702を指し示す。データ構造体を複数計算機間で共有する場合、共有したいデータは要素a、b、d、及び領域702に格納されたデータであり、要素cに格納されたアドレス値自体は共有する必要がない。これは各計算機が独立したメインメモリ106を管理しており、各計算機上での領域511内の共有データ格納アドレスは必ずしも一致しないためである。

【0057】一方、本発明に於て共有データを宣言した時に作成される共有データ複製のメインメモリ上の配置の一実施例を512に示す。512では、511で要素cを格納していた領域701に該当する703の領域には、当該データ格納領域の開始位置迄のオフセット値704を格納する。またオフセット値704で指定されるデータ格納領域の先頭705には、データ長706を格納し、それに連続してデータ702を格納する。これにより、ポインタ変数を要素とするデータ構造体を複数計算機間で共有する場合にも、共有の対象となるデータはメモリ上で連続した領域に格納することができる。

【0058】511に於ける共有データの配置と512に於ける共有データ複製との配置の対応を共有データ管理テーブル405に於いて管理している。領域511と512に格納されたデータの対応付けは、対応付けは共有データ番号をキーとして行っている。この対応は、左記に図5で示したのと同様に、アプリケーションが511に於ける共有データを使用すると宣言した時に共有データ管理テーブル405内に自動的に作成する。これによって、不連続な共有データを一つの識別子(共有データ番号)で指定でき、複数計算機間で共有する場合にネットワークを介した通信の頻度を減らすことができる。

【0059】図8は、本発明に於ける共有データ管理テ

ーブル405の他の実施例を示し、特にデータ構造体の共有、及び階層関係を持つ共有データの管理について示す図である。

【0060】本実施例に於ては、図6に示した共有データ管理テーブル405に対して、610の構造体定義エントリ、及び611の構造体サイズを示すエントリを設けている。共有データ番号601に対応する共有データがデータ構造体の場合、構造体定義エントリ610にはデータ構造体の定義を格納し、その共有データ複製領域512上でのデータ長を611に格納する。本図に於ては、共有データ番号「11」の共有データが、データ構造体であることを示している。

【0061】共有データ管理テーブル405の構造体定義エントリ610に格納する構造体定義テーブルの一実施例を図9(a)に示す。構造体定義テーブル801は、当該データ構造体の要素の型情報を格納するエントリ802、当該要素のサイズを格納するエントリ803、当該要素の当該データ構造体内での開始位置を格納するエントリ804、当該要素がポインタ型の場合に、それが指し示すデータのサイズを格納するデータサイズ・エントリ805により構成される。

【0062】更に図8に於て、階層要素フラグ・エントリ612、及び階層要素共有データ番号エントリ613を設けることにより、複数の共有データに対して階層関係を定義しておき、共有データの一貫性保証処理のための獲得処理等を下位階層として定義されたデータを含めて、一纏めで実行することが可能となる。例えば図8に於ては、共有データ番号「5」の共有データの階層データとして、共有データ番号「6」の共有データが定義されており、更に共有データ番号「6」の下位階層には何も定義されていない。この場合、例えば共有データ番号「5」の一貫性保証処理を指示すると、共有データ番号「6」の共有データに関しても同様の処理を実行する。また共有データ番号「6」の一貫性保証処理を指示した場合には、その共有データに対してのみ、同処理を実行する。

【0063】尚、階層関係にある共有データは、共有データ複製領域512上で連続して配置しておくことによって、一貫性保証処理を実行する際に一括してネットワークにデータを送出することが可能となり、ネットワークの利用効率を向上することができる。

【0064】また、図7のポインタ変数を含むデータ構造体を階層構造として管理することもできる。即ち、データ構造体の要素aからdの本体部分を一つの共有単位として扱い、その下位階層の共有単位として、要素cが指し示すデータ部分702を定義すれば良い。また複数のポインタ変数がデータ構造体内に定義されている場合には、例えば構造体定義に表れる順番で、下位階層の共有データ番号を階層要素共有データ番号エントリ613に登録すれば良い。

【0065】ポインタ変数を含むデータ構造体を階層構造として管理する別の実施例を、図9(b)に示す。構造体定義テーブル802は、801と同等の803から805のエントリと、当該要素がポインタ型の場合に、それが指し示すデータの共有データ番号を格納するエントリ806により構成される。

【0066】図10は、本発明による共有データ管理システム204が使用するバケット構造の一実施例を示す図である。

【0067】バケット901は、バケットヘッダ902とデータ903より構成される。バケットヘッダ902は、送信先計算機904、送信元計算機905、共有データ番号906、共有データ管理コマンド907より構成される。例えば、共有データ番号12を持つ共有データの変更を計算機1が計算機2に通知する場合には、送信先計算機904に「2」、送信元計算機905に1、共有データ番号906に12、共有データ管理コマンド907に「共有データ変更通知」を示す識別子を、データ903に変更された共有データの内容を格納して、共有データを保持する計算機もしくは全ての計算機にバケ

ットを送信する。

【0068】図11は、本発明による共有データ管理方式を適用するためにアプリケーション201に提供する共有データへのアクセス関数の一実施例を示す表である。

【0069】本実施例の「dsm_」で始まる文字列は、共有データ管理システムを使用するためにアプリケーションが呼び出すオペレーティング・システムの関数の名称である。アプリケーションが共有データにアクセスするためには初期化処理、一貫性保証処理、終了処理を行なう。

【0070】先ず、1001及び1002の初期化処理dsm_open、dsm_open2に於いて、アプリケーションが使用する共有単位を共有データ管理システムに宣言する。具体的にはあるメモリ領域に、共有データ番号を対応付ける。共有データ管理システムでは、共有単位を管理するために必要な管理情報のエントリを確保する。1001のdsm_openは、連続したメモリ領域を一つの共有単位として宣言する。1002のdsm_open2は、一定のサイズを持つデータが一定の間隔で並んでいるメモリ領域を一つの共有単位として宣言する。

【0071】次に1003及び1004の一貫性保証処理dsm_acquire、dsm_releaseに於いて、アプリケーションが共有単位として宣言したメモリ領域に対してRead/Write操作を行なう時は、一連のRead/Write操作を1003のdsm_acquireと1004のdsm_releaseとで囲むことによって、共有データ管理システムは共有単位の一貫性を保証する。

【0072】最後にアプリケーションが共有データ(共有単位)を必要としなくなった場合には、1005の終了処理dsm_closeを共有データ管理システムに発行する。この発行によって、初期化処理で確保した共有データ管理システム内の管理情報の領域を解放することができる。

【0073】尚、初期化処理、一貫性保証処理、終了処理時に共有データ管理システム内で行なわれる処理内容の詳細は後述する。

【0074】図12は、本発明による分散共有メモリを利用した領域分割型並列処理を行う際のアプリケーションのメイン処理の一実施例を示すフローチャートである。

【0075】ここで各計算機上のアプリケーションは、図2及び図5に示した配列データの各要素に対して、先に図2(b)を用いて説明した様な上下左右の隣接要素と自要素の値を用いた演算処理を行う。

【0076】先ずステップ1101で、補助記憶装置などに格納された配列データからアプリケーション201(1)に必要な配列データをメインメモリ106に読み込む。ステップ1102では、アプリケーションで使用する共有データの宣言を行う。具体的には、部分配列501のメモリ領域を共有データ番号「0」、部分配列502のメモリ領域を共有データ番号「1」として宣言する。図中では、それぞれの宣言を(open 0)、(open 1)と略記したが、これはそれぞれ1001のルーチンを用いて行う。ステップ1103では、アプリケーション201(1)が取り扱う部分配列の中で、共有データ番号「0」を持つ共有データを仮想的な共有空間に反映する操作をルーチン1004を用いて実行する。図中では、(rel 0)と略記した。

【0077】次のステップ1104からステップ1108までが、アプリケーションのメインループに対応する。ステップ1104で他のアプリケーションと同期を取る。ステップ1105では、仮想的な共有空間から最新データを獲得するための操作を実行する。ここでは、移動元を共有データ「1」、移動先を同じ共有データ「1」として1003のdsm_acquire処理を実行する。図中では、この操作を(acq 1)と略記した。ステップ1106では、部分配列の全ての要素について上下左右の隣接要素と自要素を用いた演算処理を実行する。ステップ1107では、ステップ1106の結果を仮想的な共有空間に反映する操作をステップ1103と同様に実行する。ステップ1108では、計算が終了したか否かを判定し、終了していない場合にはステップ1104に戻り、ステップ1108迄の処理を繰り返す。終了した場合にはステップ1109に進む。

【0078】ステップ1109では共有データの解放処理を行う。ここでは、共有データ番号「0」を持つ共有データ、共有データ番号「1」を持つ共有データの解放

処理を、ルーチン1005を用いて行う。図中では、それぞれ(close 0)、(close 1)と略記した。

【0079】アプリケーション201(2)についても、ステップ1101からステップ1109と同様な処理を実行する。但し、担当する部分配列が異なるため、ステップ1112、ステップ1114、ステップ1116では、それらに対応するアプリケーション201(1)に於ける処理ステップとは異なる共有データに対して操作を行う。

【0080】図13は、図11で示した共有データの初期化処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【0081】ブロック1201で囲まれた処理ステップは、共有データの使用を宣言したアプリケーションを実行している計算機101で行なわれる処理内容を、ブロック1202で囲まれた処理ステップは、共有データ番号で決定される共有データを管理する計算機で行なわれる処理内容を示している。

【0082】先ずステップ1203で、アプリケーションが共有データ領域管理部404に共有データ番号と共有データのメモリ領域を指定して共有データの使用を宣言するために1001のdsm_openルーチンを起動する。処理ステップ1204では、指定された共有データ番号に基づき、共有データを管理する計算機を決定する。例えば、ネットワーク計算機システムを構成する計算機に予め一意な番号を割り当てておき、共有データ番号をネットワーク計算機システムを構成する計算機の総数で割った余りが、ある予め決定された番号と一致した計算機を管理計算機として決定してもよい。ステップ1205では、共有データ領域管理部が決定した共有データの管理計算機に対して共有データを使用することを通知する。共有データを管理する計算機では、ステップ1206に於いて、宣言された共有データがすでに使用されているか否かを確認する。使用されていない場合には、ステップ1210で、管理計算機の共有データ管理テーブル405に新しいエントリを作成する。また、共有データ保持計算機リストの初期設定を行なう。使用されている場合には、ステップ1207に於いて、共有データを使用することを宣言した計算機を609の共有データ保持計算機リストに追加し、ステップ1208で共有データ保持計算機リストに登録されている計算機に対して共有データ保持計算機リストが変更されたことを通知する。

【0083】共有データの使用を宣言した計算機では、共有データが宣言されていた場合には、ステップ1209で共有データ保持計算機リストに基づき最新の共有データの内容を獲得して、それを共有データの複製とする。宣言されていなかった場合には、ステップ1211で共有データの複製のメモリ領域を確保する。

【0084】ステップ1212では、共有データ管理テーブル内に新しいエントリを作成し、共有データアドレスや共有データ複製アドレスなどの初期設定を行なう。

【0085】図14は、図11で示した共有データの一貫性保証処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【0086】ブロック1301で囲まれた処理ステップは、最新の共有データを獲得するために1003のdsm_acquireを発行した計算機で行なわれる処理内容を示している。

【0087】先ずステップ1302ではアプリケーション内の共有データに最新の共有データを獲得するために、獲得する共有データ番号(移動元)と獲得した共有データを置くメモリ領域を示す共有データ番号(移動先)を指定して共有データ領域管理部404にdsm_acquireを発行する。

【0088】ステップ1303で獲得したい共有データ番号(移動元)に対応する共有データ複製アドレス、共有データサイズなどを共有データ管理テーブル405から共有データ番号をキーとして検索する。

【0089】次にステップ1303と同様にステップ1304では、移動先を示す共有データ番号に基づいて、共有データ番号に対応する共有データアドレス、及び共有データサイズを調べる。

【0090】ステップ1305では、移動元である共有データ複製205のメモリ領域512の内容を移動先の共有データのメモリ領域511にコピーを行なう。

【0091】ステップ1305でコピーを行う際には、共有データ管理テーブル405の分割フラグ606に「1」が格納されている場合、図5に示した領域512の格納形式から領域511の格納形式への変換を行う。これは、データをコピーする際に、各要素毎に共有データオフセット値608分のアドレス間隔でコピーを行うことによって実施する。また同様に、共有データ管理テーブル405の構造体定義エントリ610に構造体定義が、また階層要素フラグ612に「1」が格納されている場合にも、格納形式の変換を行いながら、領域512から511へのデータコピーを行う。

【0092】図15は、図11で示した共有データの一貫性保証処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【0093】ブロック1401で囲まれた処理ステップは、アプリケーションの共有データを共有データ管理システムに反映するために1004のdsm_releaseを発行した計算機で行なわれる処理内容を、ブロック1402で囲まれた処理ステップは、共有データの反映があった共有データを保持している全ての計算機で行なわれる処理内容を示している。

【0094】先ずステップ1403ではアプリケーション内の共有データ内容を共有データ管理システムに反映

10

20

30

40

50

させるために、共有データ番号を指定して共有データ領域管理部に1004のdsm_releaseを発行する。

【0095】ステップ1404では、指定された共有データ番号に対応するメモリ領域を共有データ管理テーブルを共有データ番号をキーとして検索する。更に、共有データ番号付加部が、対応するメモリ領域の内容を読みだし、指定された共有データを保持する全ての計算機に対して変更を伝達するためのバケットを作成する。

【0096】ステップ1404では、共有データのメモリ領域511の内容を共有データ複製205のメモリ領域512にコピーする。この際、図14のステップ1305で行ったコピー処理とは逆方向に処理を実施する。即ち、図5及び図7に示した領域511の格納形式から領域512の格納形式への変換を行いながら、データコピーを行う。

【0097】ステップ1406では、バケット送受信部が共有データ管理テーブル内の共有データ保持計算機リストに基づいて、共有データ番号付加部が作成したバケットを送信する。

【0098】共有データを保持している計算機では、ステップ1407に於いて、ネットワークによって伝達されてきた共有データの変更を通知するバケットをバケット送受信部で受信する。更にバケット内容を調べて、共有データの変更通知であることを知り、バケット番号識別部にバケットを渡す。

【0099】ステップ1408では、バケット番号識別部がバケット内の共有データ番号を識別する。更にその共有データ番号に基づいて共有データ管理テーブルを検索し、対応する共有メモリ複製アドレス、共有メモリサイズなどを調べる。

【0100】ステップ1409では、共有データ識別部がネットワークから到着した共有データ変更通知に基づいて対応するメモリ領域に対して書き込みを行なう。

【0101】図16は、図11で示した共有データの終了処理を行なう際、共有データ管理システムが行なう処理の一実施例を示すフローチャートである。

【0102】ブロック1501で囲まれた処理ステップは、共有データの解放を要求した計算機で行なわれる処理内容を、ブロック1502で囲まれた処理ステップは、共有データ番号で決定される共有データを管理する計算機で行なわれる処理内容を示している。

【0103】先ず、ステップ1503で、アプリケーションが共有データ領域管理部に対して共有データ番号を指定して共有データを解放するために1004のdsm_releaseを発行する。処理ステップ1504では、指定された共有データ番号に基づき、共有データを管理する計算機を決定する。ステップ1505では、共有データ領域管理部が決定した共有データの管理計算機に対して共有データを解放することを通知する。

【0104】共有データを管理する計算機では、ステップ1506に於いて、解放を宣言された共有データに対して、その共有データ番号に基づいて共有データ管理テーブルの共有データ保持計算機リストから、解放を要求した計算機を削除する。

【0105】ステップ1507に於いて、共有データ管理テーブルの共有データ保持計算機リストから解放要求のあった計算機を削除した結果、共有データ保持計算機リストが空になったか否かを確認する。計算機リストが空の場合には、ステップ1508に於いて、共有データを管理している計算機内の共有データ管理テーブル405から共有データのエントリを削除する。計算機リストが空でない場合には、ステップ1509に於いて共有データ保持計算機リストに登録されている計算機に対して共有データ保持計算機リストの変更を通知する。

【0106】共有データの解放を宣言した計算機では、ステップ1510で共有データ管理テーブル内から共有データのエントリを削除し、共有データ複製のために割り当てられていたメモリ領域を解放する。

【0107】以上、本発明の実施例について説明してきたが、最後に、再度図4を用いて、本発明の好適な実施例における目的、構成、及び効果について簡単にまとめる。

【0108】ネットワークで接続した複数の計算機で構成されるネットワーク計算機システムで、アプリケーションに対してあたかも一つのデータを共有しているかの様にみせる分散共有メモリを実現するために、アプリケーションが使用する共有データに関して、共有するデータ領域や、共有するデータ一貫性保証のタイミング等の共有データに関する情報を得て、ネットワーク間を流れるデータ転送量や転送回数を減らし、ネットワーク資源を有効に活用し、共有データへのアクセス性能を向上させる。

【0109】本発明の好適な実施例である図4においては、共有データ管理システム204は、共有データ番号識別部401、共有データ番号付加部402、バケット送受信部403、共有データ領域管理部404から構成され、共有データ管理テーブル405を保持する。

【0110】アプリケーション201が共有するデータ（共有単位）とメモリへの写像を共有データ領域管理部404に対して宣言すると、共有データとメモリとの対応を共有データ管理テーブル405に格納する。共有データの一貫性を保証するために、共有データ変更を指示した計算機と共有データ変更を受け取る計算機に於いて、次の様な構成を取る。

【0111】共有データの変更を指示した計算機では、変更した共有データを他の計算機に通知するために共有データ番号に対応するメモリ領域を調べ、メモリ領域の内容に共有データ番号を付加して、通知用バケットの作成を行なう共有データ番号付加部402を設ける。

【0112】共有データ変更を受け取る計算機では、ネットワークから伝達されてきたバケットの共有データ番号を識別し、共有データ番号に対応するメモリ領域を共有データ管理部を利用して調べ、対応するメモリ領域に対してデータを書き込む処理を行なう共有データ番号識別部401を設ける。

【0113】また両計算機に於いてバケットの送受信を行ないネットワーク・インタフェースの制御を行なうバケット送受信部403を設ける。

【0114】本発明の好適な実施例によれば、アプリケーションが使用する共有データに関する情報を共有データ管理システムに通知する手段を設けたことにより、ネットワークを流れる無駄なデータ転送量や転送回数が減り、ネットワーク資源を有効に利用することができると共に、無駄なデータを転送しないのでネットワークの転送時間も短くなり共有データへのアクセス性能が向上し、例えば領域分割型の並列処理に於て、並列化による高速化率を向上することができる。

【0115】

【発明の効果】本発明によれば、ネットワークで接続した複数の計算機システムで構成されるネットワーク計算機システムに於いて、アプリケーションに対してあたかも一つのデータを共有しているかの様にみせる分散共有メモリを実現する際に、共有データの一貫性を保証するために必要なネットワークの転送量や転送回数が減少するので、ネットワーク資源を有効に活用することができる。更に、無駄なデータ転送をしないのでネットワークの転送時間が短くなり、その結果、共有データへのアクセス性能を向上させることができる。

【図面の簡単な説明】

【図1】本発明の一実施例のネットワーク計算機システムのブロック図である。

【図2】本発明の適用対象の一例である領域分割型並列処理の一実施例を示す図である。

【図3】本発明による共有データ保持の一実施例を示す図である。

【図4】本発明に於ける共有データ管理システムの一実施例を示すブロック図である。

【図5】本発明による共有データの領域分割型並列処理への応用を示す一実施例である。

【図6】本発明に於ける共有データ管理テーブルの一実施例を示す図である。

【図7】本発明による共有データのデータ構造体への応用を示す一実施例である。

【図8】本発明に於ける共有データ管理テーブルの一実施例を示す図である。

【図9】本発明に於ける構造体定義テーブルの一実施例を示す図である。

【図10】本発明に於ける通信バケット構造の一実施例を示す図である。

【図11】本発明による共有データアクセス関数の一実施例を示す図である。

【図12】本発明による共有データを利用したアプリケーションのメイン処理の一実施例を示すフローチャートである。

【図13】本発明による共有データ使用開始処理の一実施例を示すフローチャートである。

【図14】本発明による共有データ獲得処理の一実施例を示すフローチャートである。

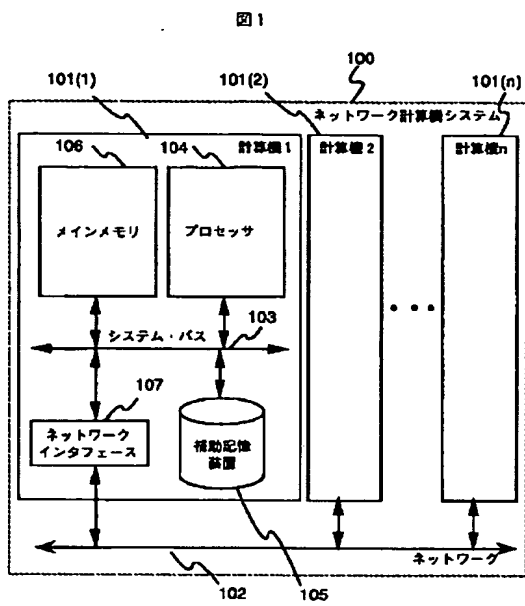
【図15】本発明による共有データ反映処理の一実施例を示すフローチャートである。

【図16】本発明による共有データ解放処理の一実施例を示すフローチャートである。

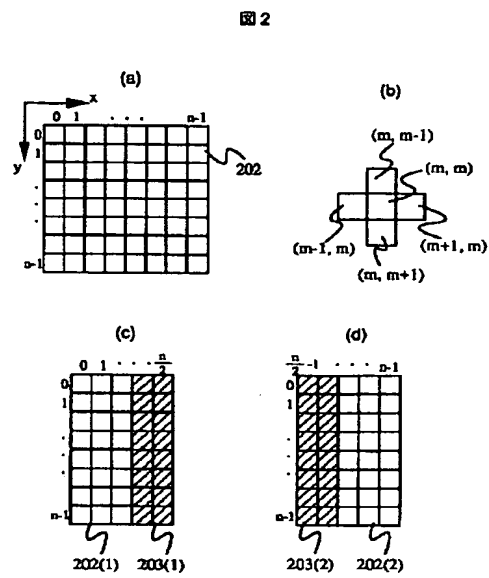
【符号の説明】

100：ネットワーク計算機システム、101：計算機、102：ネットワーク、103：システム・バス、104：プロセッサ、105：補助記憶装置、106：メインメモリ、107：ネットワーク・インタフェース、201：アプリケーション、202：アプリケーション内のローカルデータ、203：アプリケーション内の共有データ、204：共有データ管理システム、205：共有データ管理システム内の共有データ複製、401：共有データ番号識別部、402：共有データ番号付加部、403：バケット送受信部、404：共有データ領域管理部、405：共有データ管理テーブル、501：共有データ番号「0」を持つ共有データ、502：共有データ番号「1」を持つ共有データ、511：共有データのメインメモリ上での配置、512：共有データ複製のメインメモリ上での配置、601：共有データ番号、602-613：共有データ管理テーブルの管理項目、701-706：データ構造体の共有データ複製領域上での配置要素、801、802：構造体定義テーブル、803-807：構造体定義テーブルの管理項目、901：通信バケット、902-907：通信バケット構成要素、1001-1006：共有データのアクセス関数、1301-1318：共有データを利用した領域分割型アプリケーションのメイン処理ステップ、1201：共有データの使用を要求した計算機内での処理ステップ、1202：共有データ管理計算機内での処理ステップ、1203-1212：共有データ使用を要求した時の処理ステップ、1301：共有データの獲得処理を発行した計算機内での処理ステップ、1302-1305：共有データを獲得する時の処理ステップ、1401：共有データ反映処理を発行した計算機内での処理ステップ、1402：データを共有する他の計算機内での処理ステップ、1403-1410：共有データを反映する時の処理ステップ、1501：共有データ解放を要求した計算機内での処理ステップ、1502：共有データ管理計算機内での処理ステップ、1503-1510：共有データ解放を要求した時の処理ステップ。

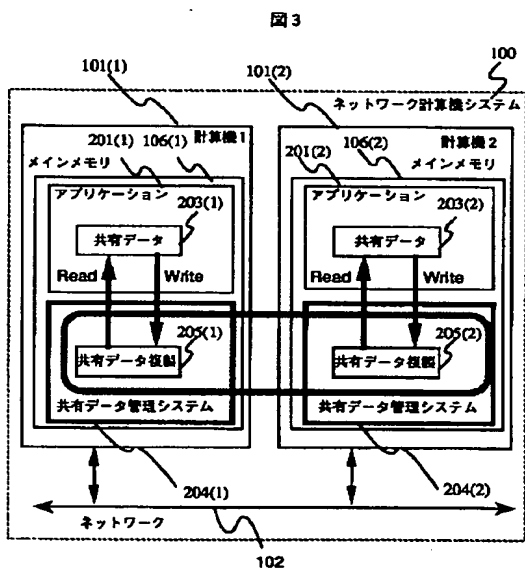
【図1】



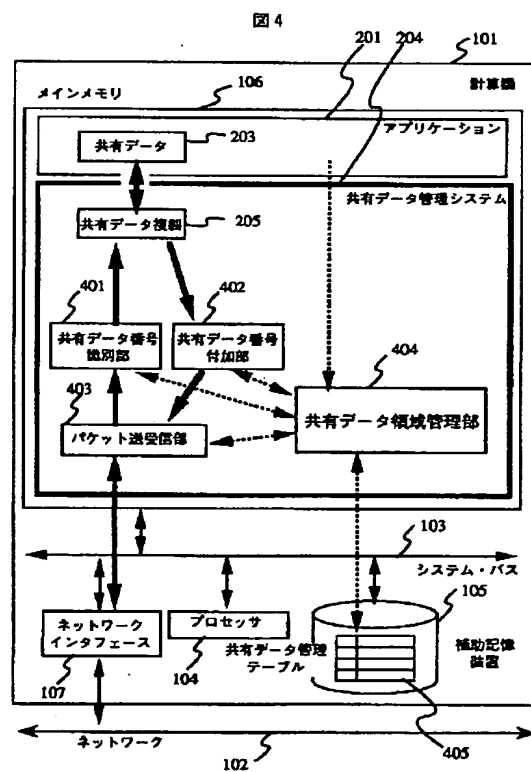
【図2】



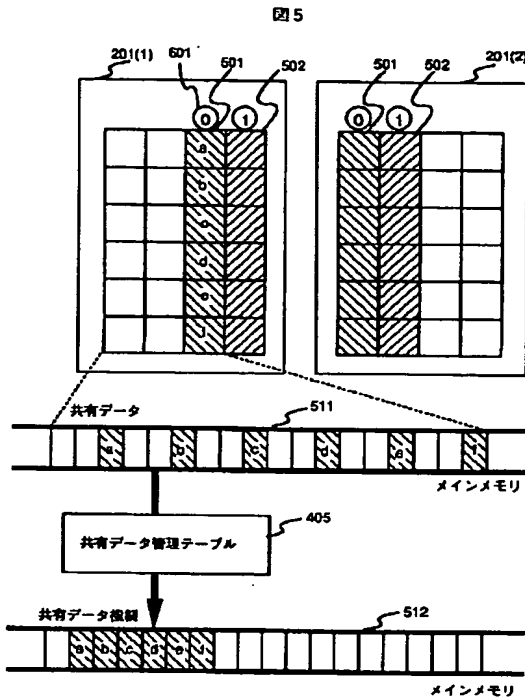
【図3】



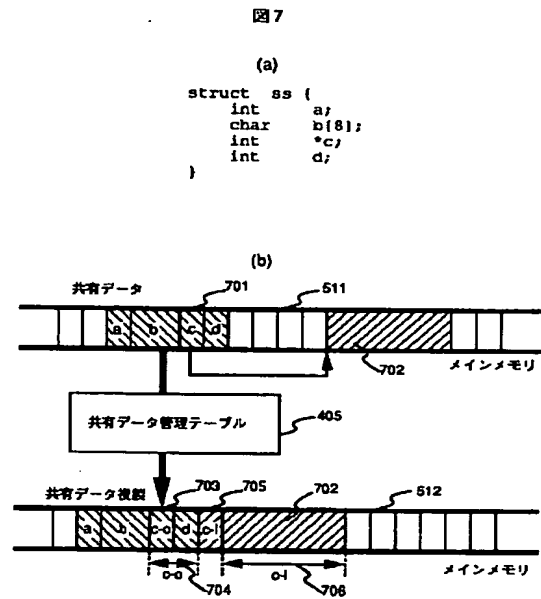
【図4】



【図5】



【図7】



【図6】

405

共有データ番号	管理ノードフラグ	共有データアドレス	共有データ領域アドレス	共有データサイズ	分配フラグ	共有データ要素数	共有データオフセット	共有データ保持計算機リスト
1	1	101234	1234	2000	0	—	—	1,2,3
5	0	108768	8768	1024	0	—	—	4,5
11	0	104892	4892	8	1	10	80	7,8,9,10
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	1	107485	7485	8	1	20	24	1,2,3,4,5,6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

601 602 603 604 605 606 607 608 609

図6

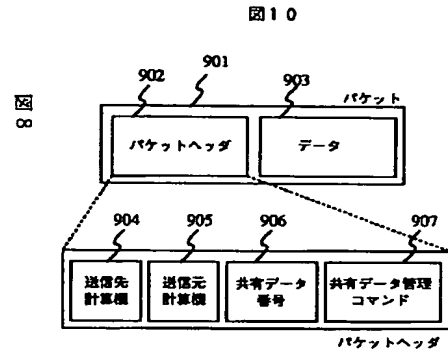
【図8】

405

共有データ番号		共有データ保持計算機リスト	構造体定義	構造体サイズ	搬送要素フラグ	搬送要素共有データ番号
1		1,2,3	0	0	0	0
5		4,5	0	0	1	6
11	●●●	7,8,9,10	struct ss	40	0	0
●		●	●	●	●	●
●		●	●	●	●	●
6		1,2,3,4,5,6	0	0	0	0
●		●	●	●	●	●
●		●	●	●	●	●

601 602~608 609 610 611 612 613

【図10】



【図9】

図9

(a)

要素型	要素サイズ	開始位置	データサイズ
int	4	0	
char[5]	8	4	
int *	4	12	16
int	4	16	

803 804 805 806

(b)

要素型	要素サイズ	開始位置	共有データ番号
int	4	0	
char[5]	8	4	
int *	4	12	10
int	4	16	

803 804 805 807

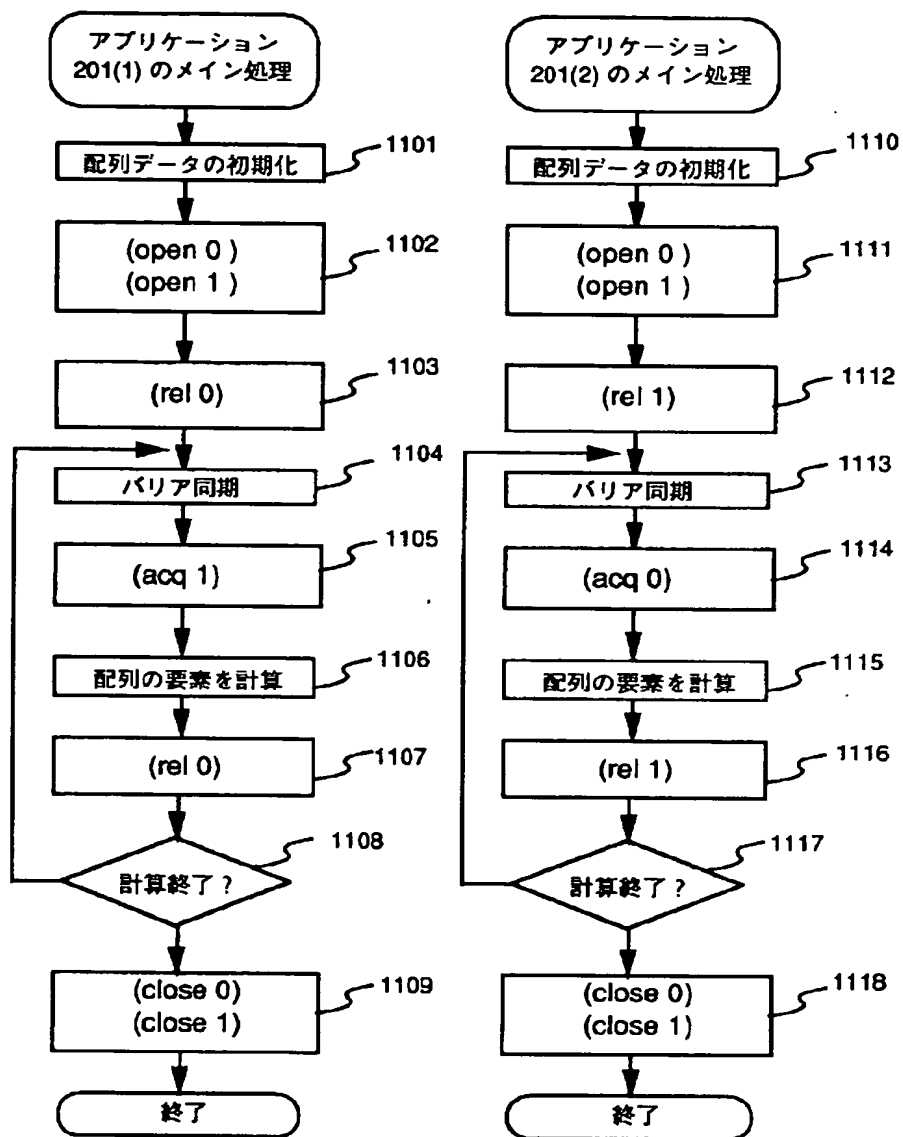
【図11】

図11

初期化処理	dsm_open	(共有データ番号、先頭アドレス、領域サイズ)	1001
	dsm_open2	(共有データ番号、先頭アドレス、要素サイズ、要素数、オフセットサイズ)	1002
一貫性保証処理	dsm_acquire	(移動元共有データ番号、移動先共有データ番号)	1003
	dsm_release	(共有データ番号)	1004
終了処理	dsm_close	(共有データ番号)	1005

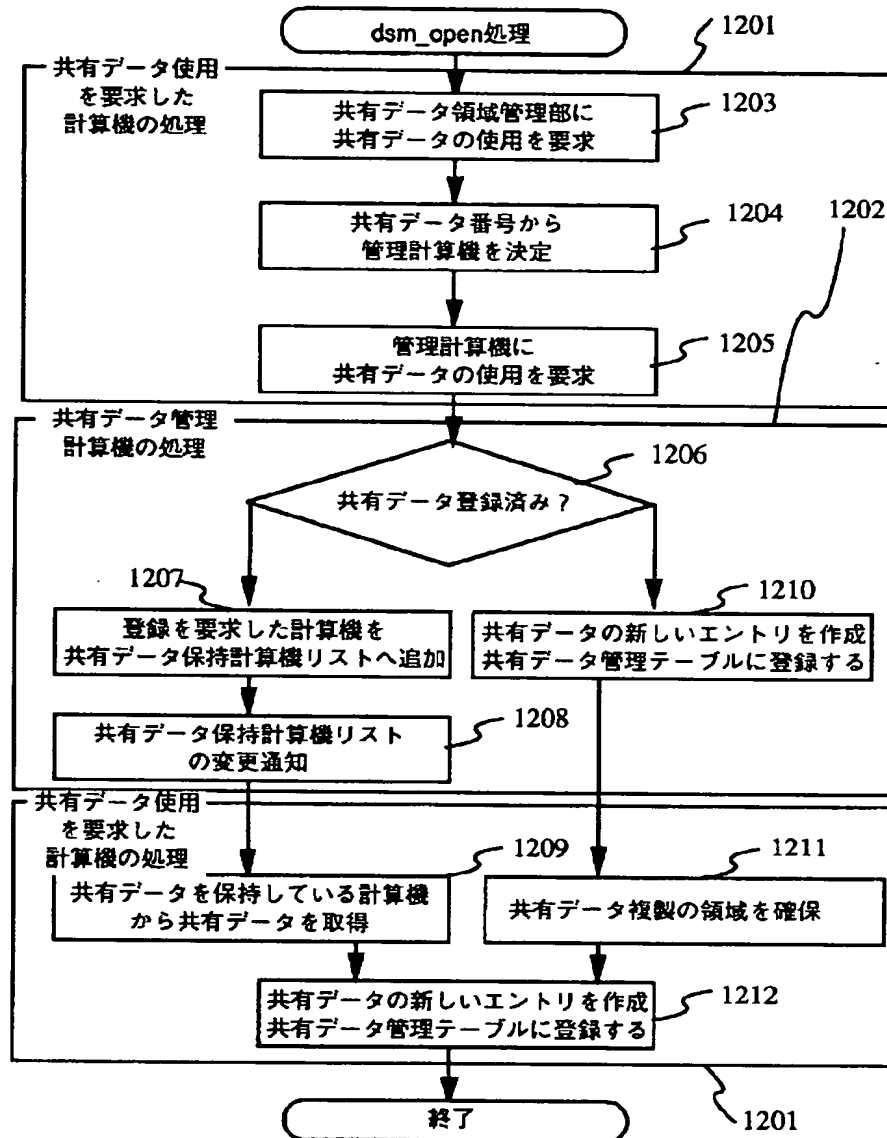
【図12】

図12

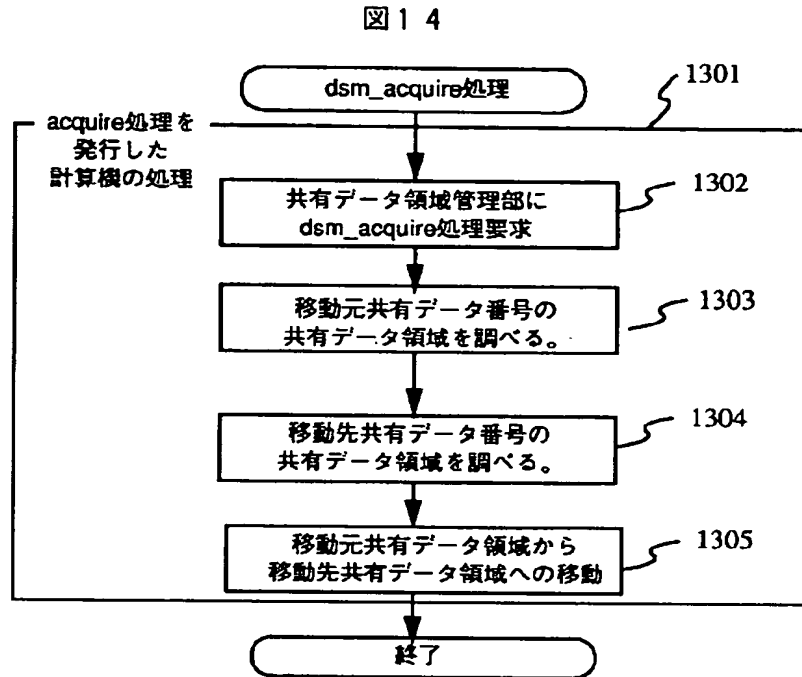


【図13】

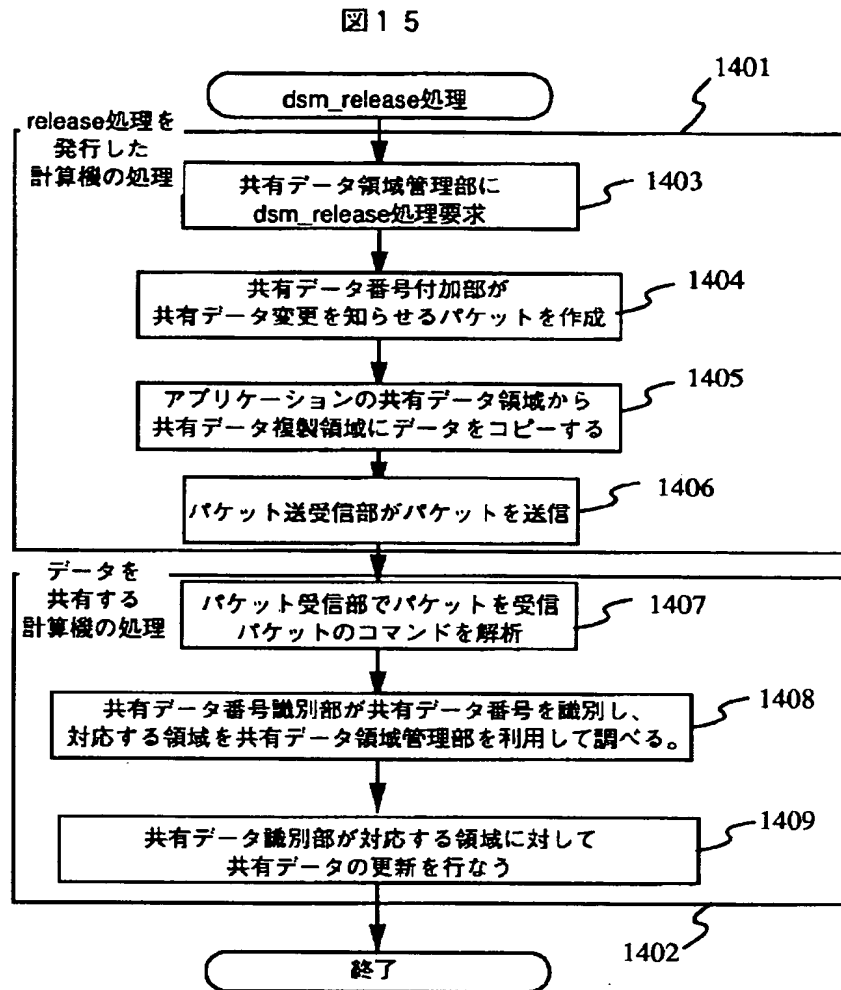
図13



【図14】

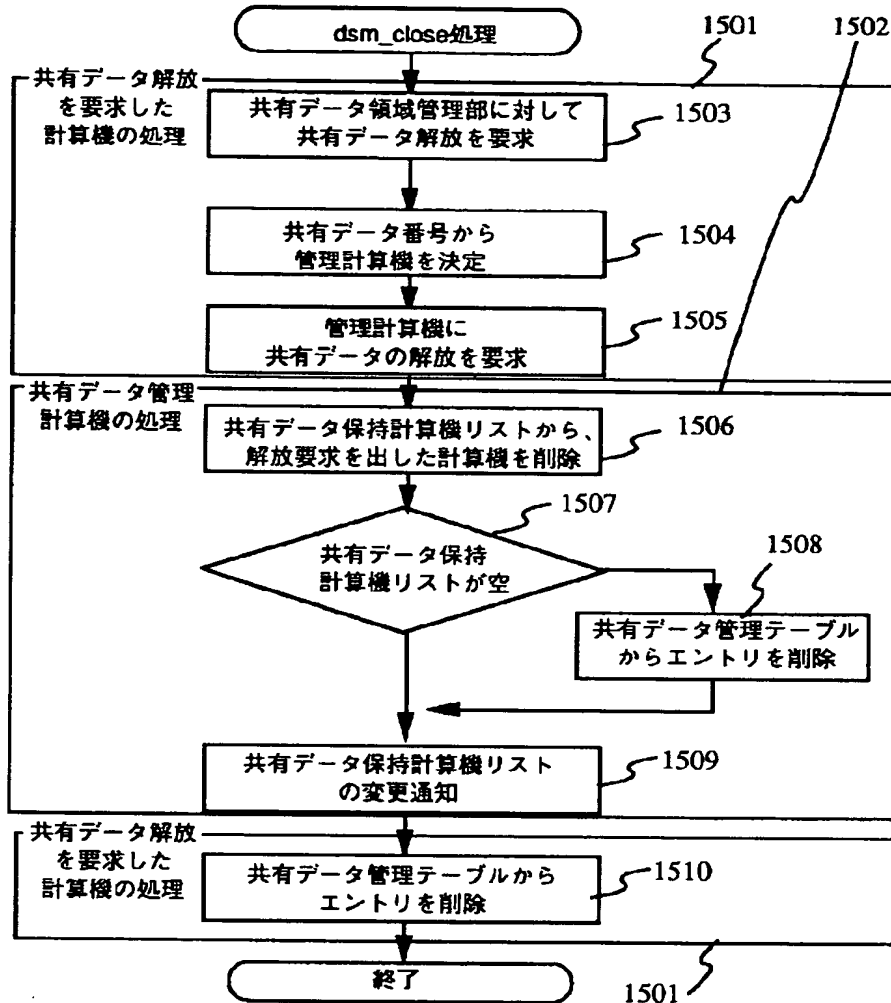


【図15】



【図16】

図16



フロントページの続き

(72)発明者 林 剛久
 東京都国分寺市東恋ヶ窪1丁目280番地
 株式会社日立製作所中央研究所内

(72)発明者 鬼頭 昭
 神奈川県横浜市戸塚区戸塚町5030番地 株
 式会社日立製作所ソフトウェア開発本部内